

BREVET D'INVENTION

CERTIFICAT D'UTILITÉ - CERTIFICAT D'ADDITION

COPIE OFFICIELLE

Le Directeur général de l'Institut national de la propriété industrielle certifie que le document ci-annexé est la copie certifiée conforme d'une demande de titre de propriété industrielle déposée à l'Institut.

Fait à Paris, le 07 OCT. 2004

Pour le Directeur général de l'Institut
national de la propriété industrielle
Le Chef du Département des brevets

**PRIORITY
DOCUMENT**
SUBMITTED OR TRANSMITTED IN
COMPLIANCE WITH RULE 17.1(a) OR (b)

Martine PLANCHE

BEST AVAILABLE COPY

INSTITUT
NATIONAL DE
LA PROPRIÉTÉ
INDUSTRIELLE

SIEGE
26 bis, rue de Saint-Petersbourg
75800 PARIS cedex 08
Téléphone : 33 (0)1 53 04 53 04
Télécopie : 33 (0)1 53 04 45 23
www.inpi.fr



26 bis, rue de Saint Pétersbourg
75800 Paris Cedex 08
Téléphone : 33 (1) 53 04 53 04 Télécopie : 33 (1) 42 94 86 54

BREVET D'INVENTION CERTIFICAT D'UTILITÉ

Code de la propriété Intellectuelle - Livre VI



REQUÊTE EN DÉLIVRANCE

page 1/2

BR1

Cet imprimé est à remplir lisiblement à l'encre noire

DB 540 • 9 / 210502

REMISE DES PIÈCES DATE 30 SEPT 2003 LIEU 75 INPI PARIS N° D'ENREGISTREMENT 0311434 NATIONAL ATTRIBUÉ PAR L'INPI DATE DE DÉPÔT ATTRIBUÉE 30 SEP. 2003 PAR L'INPI		1 NOM ET ADRESSE DU DEMANDEUR OU DU MANDATAIRE À QUI LA CORRESPONDANCE DOIT ÊTRE ADRESSÉE SANTARELLI 14 Avenue de la Grande Armée 75017 PARIS	
Vos références pour ce dossier (facultatif) BIF116008/ON/LJH			
Confirmation d'un dépôt par télécopie		<input type="checkbox"/> N° attribué par l'INPI à la télécopie	
2 NATURE DE LA DEMANDE		Cochez l'une des 4 cases suivantes	
Demande de brevet		<input checked="" type="checkbox"/>	
Demande de certificat d'utilité		<input type="checkbox"/>	
Demande divisionnaire		<input type="checkbox"/>	
Demande de brevet initiale		N° _____ Date _____	
ou demande de certificat d'utilité initiale		N° _____ Date _____	
Transformation d'une demande de brevet européen		<input type="checkbox"/>	
Demande de brevet initiale		N° _____ Date _____	
3 TITRE DE L'INVENTION (200 caractères ou espaces maximum) Procédé et dispositif d'édition de documents graphiques numériques du type SVG notamment à partir d'un butineur.			
4 DÉCLARATION DE PRIORITÉ OU REQUÊTE DU BÉNÉFICE DE LA DATE DE DÉPÔT D'UNE DEMANDE ANTÉRIEURE FRANÇAISE		Pays ou organisation _____ N° _____ Date _____ Pays ou organisation _____ N° _____ Date _____ Pays ou organisation _____ N° _____ <input type="checkbox"/> S'il y a d'autres priorités, cochez la case et utilisez l'imprimé «Suite»	
5 DEMANDEUR (Cochez l'une des 2 cases)		<input checked="" type="checkbox"/> Personne morale <input type="checkbox"/> Personne physique	
Nom ou dénomination sociale		CANON KABUSHIKI KAISHA	
Prénoms			
Forme juridique		Société de droit japonais	
N° SIREN		_____	
Code APE-NAF		_____	
Domicile ou siège	Rue	3-30-2, Shimomaruko, 3-chome, Ohta-ku,	
	Code postal et ville	_____ Tokyo	
	Pays	JAPON	
Nationalité		JAPONAISE	
N° de téléphone (facultatif)		N° de télécopie (facultatif)	
Adresse électronique (facultatif)			
<input type="checkbox"/> S'il y a plus d'un demandeur, cochez la case et utilisez l'imprimé «Suite»			

Remplir impérativement la 2^{ème} page

**BREVET D'INVENTION
CERTIFICAT D'UTILITÉ**

REQUÊTE EN DÉLIVRANCE
page 2/2

BR2

REMISE EN DÉPÔT **30 SEPT 2003** Réservé à l'INPI
DATE **75 INPI PARIS**
UEU **0311434**
N° D'ENREGISTREMENT
NATIONAL ATTRIBUÉ PAR L'INPI

08 540 W / 210502

6 MANDATAIRE (s'il y a lieu)		
Nom		
Prénom		
Cabinet ou Société	SANTARELLI	
N° de pouvoir permanent et/ou de lien contractuel		
Adresse	Rue	14 Avenue de la Grande Armée
	Code postal et ville	75 017 Paris
	Pays	FRANCE
N° de téléphone (facultatif)		
N° de télécopie (facultatif)		
Adresse électronique (facultatif)		
7 INVENTEUR (S)		Les inventeurs sont nécessairement des personnes physiques
Les demandeurs et les inventeurs sont les mêmes personnes		<input type="checkbox"/> Oui <input checked="" type="checkbox"/> Non : Dans ce cas remplir le formulaire de Désignation d'inventeur(s)
8 RAPPORT DE RECHERCHE		Uniquement pour une demande de brevet (y compris division et transformation)
Établissement immédiat ou établissement différé		<input checked="" type="checkbox"/> <input type="checkbox"/>
Paiement échelonné de la redevance (en deux versements)		Uniquement pour les personnes physiques effectuant elles-mêmes leur propre dépôt <input type="checkbox"/> Oui <input type="checkbox"/> Non
9 RÉDUCTION DU TAUX DES REDEVANCES		Uniquement pour les personnes physiques <input type="checkbox"/> Requête pour la première fois pour cette invention (joindre un avis de non-imposition) <input type="checkbox"/> Obtenue antérieurement à ce dépôt pour cette invention (joindre une copie de la décision d'admission à l'assistance gratuite ou indiquer sa référence): AG <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
10 SÉQUENCES DE NUCLEOTIDES ET/OU D'ACIDES AMINÉS		<input type="checkbox"/> Cochez la case si la description contient une liste de séquences
Le support électronique de données est joint		<input type="checkbox"/>
La déclaration de conformité de la liste de séquences sur support papier avec le support électronique de données est jointe		<input type="checkbox"/>
Si vous avez utilisé l'imprimé «Suite», indiquez le nombre de pages jointes		
11 SIGNATURE DU DEMANDEUR OU DU MANDATAIRE (Nom et qualité du signataire) Olivier NICOLLE N°92.3040 SANTARELLI		VISA DE LA PRÉFECTURE OU DE L'INPI M. ROCHET

La présente invention concerne l'édition de documents graphiques numériques du type SVG ou analogue, notamment à partir d'un butineur.

Le langage SVG, acronyme anglo-saxon pour "*Scalable Vector Graphic*", est un langage de description de documents graphiques bi-
5 dimensionnels écrits en langage de balisage de type XML (eXtensible Markup Language) ou analogue.

Le langage SVG admet trois types de documents ou objets graphiques : des contours graphiques vectoriels (par exemple, des tracés consistant en lignes droites et courbes), des images et du texte. Les objets
10 graphiques peuvent être regroupés, transformés et composés dans des objets précédemment rendus. Le langage SVG comprend un ensemble de fonctions telles que des transformations imbriquées, des tracés de rognage, des masques, et des objets de gabarit.

Les objets SVG peuvent être interactifs et/ou dynamiques. On peut
15 ainsi définir et déclencher des animations, soit en incorporant des éléments d'animation SVG dans un contenu SVG, soit via un script comprenant par exemple des commandes exécutables.

On connaît déjà des éditeurs qui permettent de modifier du contenu Web, telles que des pages Web écrites en langage de balisage de type HTML
20 ("*HyperText Markup Language*"), directement à partir d'un simple butineur Internet, appelé encore "browser" en anglo-saxon.

Le Demandeur s'est posé le problème de modifier de la même façon que pour le texte des documents graphiques numériques tels que des objets graphiques SVG publiés sur le réseau Internet Web à l'aide d'un simple
25 butineur Internet.

Le langage SVG étant un format textuel en raison de son appartenance aux langages de balisage, il est possible de modifier tout document SVG à partir d'un simple éditeur texte.

Cependant, le Demandeur souhaite pouvoir offrir un environnement
30 plus convivial et interactif, par exemple de type WYSIWYG ("*What you see is what you get*"), c'est-à-dire "ce que vous attrapez est ce que vous voyez" où les interactions se font notamment à l'aide d'un dispositif d'interface homme machine du type souris ou analogue.



La présente invention apporte justement une solution à ce problème.

Elle vise ainsi à fournir un éditeur de documents graphiques numériques de type SVG publiés sur le Web, apte à être exécuté sur n'importe quel butineur capable d'afficher des documents SVG en mode lecture.

Elle porte sur un procédé de traitement d'au moins un document graphique numérique représenté en un langage de balisage prédéterminé dans lequel au moins un logiciel d'affichage de type butineur est utilisé pour afficher un tel document.

Selon une définition générale de l'invention, le procédé comprend les étapes suivantes :

-i) transformer le document d'origine affiché en mode lecture en une version éditable dans le langage de balisage selon un jeu de règles de transformation prédéfinies non liées au document qui incorporent des règles d'écriture au document,

-ii) interagir via le logiciel d'affichage avec ladite version éditable pour la modifier selon ledit jeu de règles d'écriture, et

-iii) transformer la version éditable ainsi modifiée en une version en mode lecture incorporant les modifications apportées lors de l'étape ii).

Selon une réalisation, le procédé comprend en outre l'étape iv) dans laquelle il est prévu d'afficher ou stocker ladite version modifiée en mode lecture.

En pratique, le jeu de règles de transformation appartient au groupe formé par des scripts du type Javascript et XSLT. Ces règles peuvent aussi être mise sous la forme d'un service web, prenant un document SVG non éditable en entrée et renvoyant un document SVG éditable en sortie.

En pratique, le jeu de règles d'écriture appartient au groupe formé par des fonctions d'insertion, de modification, d'élimination, de déplacement, et de copie d'au moins certains éléments ou nœuds du document à éditer.

Selon une autre réalisation, la transformation inverse selon l'étape iii) est apte, à partir d'un document édité, à récupérer le document non édité.

Par exemple, la transformation directe selon l'étape i) est apte à ajouter des informations de guidage susceptibles de guider la transformation inverse selon l'étape iii).

5 En pratique, les informations de guidage appartiennent au groupe formé par des éléments à éliminer ; des éléments se trouvant dans le document modifié (ISVG2) dans un espace de nommage spécifique ; des scripts mettant à jour les valeurs des informations de guidage ; des informations d'indication relatives à la création/modification des attributs.

10 Selon encore une autre réalisation, la transformation directe selon l'étape i) est apte à identifier chaque élément graphique sélectionnable.

Selon encore une autre réalisation, la transformation directe selon l'étape i) est apte à englober/faire passer des nœuds écrits dans le langage de balisage de type SVG dans un espace de nommage non SVG pour désactiver le effets desdits nœuds.

15 Selon une autre caractéristique du procédé selon l'invention, la transformation inverse selon l'étape iii) est apte à récupérer les nœuds SVG passés dans un espace de nommage non SVG.

20 Par exemple, la transformation directe selon l'étape i) comprend un paramètre susceptible de décider de conserver/d'éliminer un élément d'animation.

Selon un autre exemple, la transformation directe selon l'étape i) comprend des événements de mutation aptes à modifier de façon synchrone le document par rapport au document initial.

25 Selon encore une autre réalisation, la transformation directe selon l'étape i) est apte à visualiser dynamiquement le document édité.

En pratique, la transformation directe selon l'étape i) incorpore dans les règles d'écriture un mécanisme apte à modifier tout ou partie du document édité via des programmes disponibles à distance du document.

30 De préférence, lesdits programmes appartiennent au groupe formé par des scripts JavaScript ; des scripts XSLT ; des services Web.

Par exemple, lesdits programmes sont aptes à prendre en entrée un arbre écrit en langage de balisage de type SVG et retourner une image non vectorielle en sortie.

En pratique, la transformation inverse selon l'étape iii) est apte à modifier ou éliminer un script d'initialisation pour sauvegarder des modifications effectuées sur des éléments graphiques créés par ledit script d'initialisation.

5 La présente invention a également pour objet un dispositif de traitement d'au moins un document graphique numérique représenté en un langage de balisage prédéterminé dans lequel au moins un logiciel d'affichage de type butineur est utilisé pour afficher un tel document.

Selon un autre aspect de l'invention, le dispositif comprend :

- 10 - des moyens de transformation directe pour transformer le document d'origine affiché en mode lecture en une version éditable dans le langage de balisage selon un jeu de règles de transformation spécifiques non liées au document qui incorporent des règles d'écriture au document;
- des moyens de traitement pour interagir via le logiciel d'affichage avec ladite version éditable pour la modifier selon ledit jeu de règles d'écriture ; et
- 15 - des moyens de transformation inverse pour transformer la version éditable ainsi modifiée en une version en mode lecture incorporant les modifications ainsi apportées par les moyens de traitement.

Selon une réalisation, le dispositif comprend en outre des moyens pour afficher ou stocker ladite version modifiée en mode lecture.

20 En pratique, le jeu de règles d'écriture appartient au groupe formé par des fonctions d'insertion, de modification, d'élimination, de déplacement, et de copie d'au moins certains éléments ou nœuds du document à éditer.

Selon une réalisation, les moyens de transformation inverse sont aptes à récupérer à partir d'un document édité, le document non édité.

25 Selon une autre réalisation, les moyens de transformation directe sont aptes à ajouter des informations de guidage susceptibles de guider les moyens de transformation inverse.

30 En pratique, les informations de guidage appartiennent au groupe formé par des éléments à éliminer ; des éléments se trouvant dans le document modifié dans un espace de nommage spécifique ; des scripts mettant à jour les valeurs des informations de guidage ; des informations d'indication relatives à la création/modification des attributs.

Selon encore une autre réalisation, les moyens de transformation directe sont aptes à identifier chaque élément graphique sélectionnable.

5 En pratique, les moyens de transformation directe sont aptes à englober/faire passer des nœuds écrits dans le langage de balisage de type SVG dans un espace de nommage non SVG pour désactiver le effets desdits nœuds.

Par exemple, les moyens de transformation inverse sont aptes à récupérer les nœuds SVG passés dans un espace de nommage non SVG.

10 De préférence, les moyens de transformation directe sont adaptés pour utiliser un paramètre susceptible de décider de conserver/d'éliminer un élément d'animation.

Selon encore une autre réalisation, les moyens de transformation directe sont adaptés pour traiter des évènements de mutation aptes à modifier de façon synchrone le document par rapport au document initial.

15 En pratique, les moyens de transformation directe sont aptes à visualiser dynamiquement le document édité.

20 Selon encore une autre réalisation, les moyens de transformation directe incorporent aux règles d'écriture un mécanisme apte à modifier tout ou partie du document édité, via des programmes disponibles à distance du document.

Par exemple, lesdits programmes appartiennent à au groupe formé par des scripts JavaScript ; des scripts XSLT ; des services Web.

25 De même, lesdits programmes sont aptes à prendre en entrée un arbre écrit en langage de balisage de type SVG et retourner une image non vectorielle en sortie.

30 La présente invention a également pour objet un support d'informations lisible par un système informatique, éventuellement amovible, totalement ou partiellement, notamment CD-ROM ou support magnétique, tel un disque dur ou une disquette, ou support transmissible, tel un signal électrique ou optique, caractérisé en ce qu'il comporte des instructions d'un programme d'ordinateur permettant la mise en œuvre du procédé visé ci-avant, lorsque ce programme est chargé et exécuté par un système informatique.



5 La présente invention a également pour objet un programme d'ordinateur stocké sur un support d'informations, ledit programme comportant des instructions permettant la mise en œuvre d'un procédé de traitement visé ci-avant, lorsque le programme est chargé et exécuté par un système informatique.

D'autres caractéristiques et avantages de l'invention apparaîtront à la lumière de la description détaillée ci-après et des dessins dans lesquels :

-la figure 1 est un organigramme illustrant les étapes principales du procédé d'édition selon l'invention ;

10 -la figure 2 est un organigramme illustrant l'étape de transformation selon l'invention ;

-la figure 3 est un organigramme illustrant l'étape de traitement SVG selon l'invention ;

15 - la figure 4 est un organigramme illustrant l'étape de transformation de nœuds scripts selon l'invention ;

- la figure 5 est un organigramme illustrant l'étape de transformation inverse selon l'invention ; et

- la figure 6 représente schématiquement un dispositif permettant de mettre en œuvre le procédé selon l'invention.

20 En référence à la **figure 1**, le procédé d'édition selon l'invention comprend au moins cinq étapes principales.

En premier lieu, selon l'étape E100, il est prévu d'acquérir une image ISVG1 sur le réseau Internet ou toile "Web".

25 En second lieu, selon l'étape E110, il est prévu de transformer cette première image ISVG1 en une deuxième image ISVG2. Cette deuxième image ISVG2 correspond à une version éditable de la première image ISVG1, selon un jeu de règles d'écriture prédéfinies non liées à l'image ISVG1.

En troisième lieu, selon l'étape E120, il est prévu de lire la deuxième image ISVG2.

30 En quatrième lieu, selon l'étape E130, il est prévu d'interagir ou modifier la deuxième image ISVG2 selon ledit jeu de règle d'écriture.

En cinquième lieu, selon l'étape E140, il est prévu d'effectuer la transformation inverse de l'étape E110 pour obtenir une version mise à jour ISGV3 de la première image ISVG1.

5 Le cas échéant, selon l'étape E150, il est prévu de stocker la version mise à jour ISVG3 sur le serveur d'origine ou autre moyen de stockage.

Les transformations des étapes E110 et E140 sont de préférence effectuées à partir d'un simple butineur Internet.

En référence à la **figure 2**, la transformation E110 décrite en référence à la **figure 1**, s'effectue en pratique en deux parties.

10 Après l'acquisition du document ISVG1 à éditer (étape E200), on effectue une transformation (étape E210) de ce document en un nouveau document XML.

15 Ce nouveau document XML contient toutes les informations de l'image sous une nouvelle forme et on l'insère à l'étape E220 dans un élément graphique d'un document SVG. Cet élément récupère les attributs de l'élément racine <svg/> et on lui ajoute un attribut xml:base pour conserver les références de type URI relative. Cet élément est inclus dans un autre élément graphique qui permet de mettre à l'échelle voulue le document SVG édité.

20 Cet élément graphique possède par ailleurs des attributs interactifs qui peuvent être par exemple des événements souris ou des événements mutation.

En pratique, les événements souris permettent la sélection d'éléments ainsi que la modification interactive ("*drag & drop*").

25 Les événements "mutation" permettent de suivre les modifications effectuées au document SVG, ce qui est utile pour faire de la mise à jour synchrone si nécessaire, mais aussi pour mettre à jour les informations permettant de réaliser facilement la transformation inverse E140 que l'on décrira plus en détail ci-après.

30 On ajoute ensuite à ce nouveau document, des informations XML (étapes E230 à E260) pour construire un nouveau document SVG. Ces dernières informations ne sont pas spécifiques à une image et correspondent aux différentes interfaces utilisateurs permettant l'édition proprement dite.

Ces informations comprennent notamment des moyens permettant de sélectionner aisément tout élément graphique éditable (étape E230) : il s'agit par exemple d'une liste d'éléments, d'un script apte à récupérer les événements souris permettant de connaître l'élément sélectionné à la souris.

5 Selon l'étape E240, ces informations comprennent des moyens de modifier les éléments SVG édité, éventuellement en ajoutant de nouveaux éléments SVG. En pratique, il s'agit d'ajouter des éléments graphiques permettant à l'utilisateur d'entrer les paramètres de chaque élément graphique à modifier ou créer.

10 Conformément à l'étape E250, ces informations comprennent des moyens pour appliquer des traitements externes sur tout ou partie du document SVG édité. En pratique, il s'agit de récupérer une partie du document SVG et d'exécuter sur cette partie une modification définie hors de l'éditeur, sur un réseau ou un disque dur par exemple. Par exemple, il peut s'agir d'une
15 transformation XSLT, un script JavaScript, une invocation de service Web.

Conformément à l'étape E260, ces informations comprennent des moyens pour contrôler les fonctionnalités ajoutées. Par exemple, ces moyens permettent d'ajouter certaines informations, notamment pour zoomer sur une partie du document, pour passer en mode Script Actif/Script Passif, contrôler
20 l'échelle du temps.

On peut choisir d'implémenter les étapes E200 à E260 sous la forme d'un unique script. On peut aussi séparer l'implémentation en deux parties, la première correspondant à une transformation (XSLT, JavaScript...) du document SVG à éditer, la deuxième partie étant un document SVG d'édition
25 comprenant directement les données des étapes E230 à E260. La transformation du document à éditer est alors utilisée dans le document SVG d'édition.

En référence à la **figure 3**, la première transformation (E110, **figure 1**) on applique sur chaque enfant de la première balise <svg/>. On crée ainsi un ou plusieurs arbres XML, assemblé en un unique résultat utilisé à
30 l'étape E220 (**figure 2**).

La première étape E300 consiste à acquérir le nœud à traiter.

On teste à l'étape E305 s'il s'agit d'un nœud défini par référence ou directement.

5 S'il s'agit d'une référence relative extérieure au document, on la transforme en une référence absolue (ce qui permet de stocker le document édité sous n'importe quelle URI).

10 S'il s'agit d'un nœud de type script (étape E305), on transforme (étape E310) le nœud script grâce à l'algorithme décrit en référence à la **figure 4**. Sinon, on teste à l'étape E315 si le nœud a un identificateur. On lui ajoute (étape E325) un identificateur si (étape E320) l'élément est un élément graphique sans identificateur, que l'on peut visualiser et sélectionner à la souris. Ceci permet une mise en œuvre plus facile de la sélection graphique d'élément.

15 On utilise ensuite l'espace de nommage des informations de gestion. SVG ayant un modèle de contenu ouvert, toutes les informations qui ne sont pas dans l'espace de nommage SVG ne sont pas prises en compte par un lecteur SVG. On peut donc utiliser cette possibilité pour ajouter des informations non utiles pendant la phase d'interaction avec l'utilisateur mais qui sont utiles pendant la transformation inverse.

L'étape E330 teste ainsi si le nœud courant est un nœud d'animation et si les animations sont à désactiver ou pas.

20 Si c'est le cas, on passe (étape E335) le nœud dans l'espace de nommage des informations de gestion ce qui permet de désactiver l'animation pendant la phase d'interaction, de réintégrer ces animations lors de la phase de transformation, et aucune perte d'information.

25 On initialise ensuite à l'étape E340 les informations nécessaires à la transformation inverse. Ces informations (que l'on met sous la forme d'attributs) permettent de savoir lors de la transformation inverse : si les éléments ont été modifiés ou non, si les éléments sont à enlever ou non, ou si les éléments graphiques sont définis en XML ou créés/modifiés via un script utilisant DOM (représentation graphique programmatique d'un document XML ou
30 "*Document Object Model*" en anglais.

Dans le cas de la création/modification par script, ces informations permettent en outre de savoir s'il s'agit d'un script d'initialisation au moment du chargement du SVG.



Des éléments peuvent être créés par un script (au moment de la visualisation du document par exemple) auquel cas, il faut les enlever au moment de la transformation inverse. Par ailleurs, des éléments ajoutés par un script appelé dans le document SVG original ne sont pas forcément modifiables. Ces informations de gestion permettent donc de différencier l'édition de ces éléments.

D'autres informations sont aussi stockées dans l'espace de nommage de gestion : par exemple l'URI du document originale au document SVG pour permettre l'étape de stockage du fichier modifié et faciliter la transformation inverse.

Lors de l'étape E345, on teste ensuite si le nœud à traiter correspond à un nœud référence (type <symbol/>). Dans ce cas, on lui ajoute un attribut contenant la liste des éléments qui référencent ce nœud. Si l'utilisateur veut ensuite modifier un élément référençant ce nœud, on lui demande s'il souhaite modifier le nœud (et donc l'ensemble des éléments référents) ou uniquement l'élément graphique, auquel cas, on clone le nœud, on le modifie et on pointe l'élément graphique vers le nœud modifié. Enfin, si le nœud a des enfants à traiter (étape E355), on traite récursivement de la même façon chacun des nœuds (étape E360), sinon l'algorithme se termine (étape E365) en renvoyant le résultat du traitement.

Pour terminer le traitement du document SVG, on copie les attributs de l'élément racine <SVG/> dans l'espace de nommage de gestion pour permettre à la transformation inverse de les récupérer. Par ailleurs, les attributs de l'élément racine sont repris en partie dans les éléments graphiques englobant le résultat de la transformation, notamment les attributs "événement".

En référence à la **figure 4**, on a représenté le traitement des nœuds scripts.

Les effets des scripts peuvent être gênants durant la phase d'édition du SVG, on permet donc à l'utilisateur de les désactiver et activer.

Pour ce faire, après l'acquisition d'un nœud script (étape E400 qui suit l'étape E310), on récupère la valeur du nœud (étape E430) et pour chaque fonction du nœud (étapes E440, E450 et E470), on ajoute un test (étape E460) permettant de savoir si la fonction doit être exécutée ou pas.

Ce test correspond simplement à regarder si un attribut "scriptEnable" pour script actif, est mis à vrai ou faux. L'utilisateur peut modifier cet attribut lors de la phase d'édition pour activer/désactiver ces scripts.

5 Généralement, les fonctions appelées par le procédé de l'invention sont définies directement dans le document. Si ces fonctions sont définies dans un autre document, il est possible de définir le nœud script par référence au document externe. Dans le cas où un nœud script serait défini par référence (étape E410), on récupère le document pointé et on l'insère en tant que valeur de ce nœud (étape E420). On élimine par ailleurs la référence au document
10 externe.

La plupart des interactions consistent à modifier des éléments qui sont directement écrits dans le document SVG. Dans ce cas, il suffit de modifier l'élément en utilisant les fonctionnalités DOM. On met à jour les données de gestion, notamment les attributs correspondant au test de
15 modification/non-modification des nœuds. On propose aussi à l'utilisateur d'ajouter de nouveaux éléments. Toutes les fonctionnalités DOM de modification de l'arbre (changement de place/clonage de nœuds par exemple) sont possibles, pourvu qu'elles respectent la norme SVG.

Il est possible que le document SVG contienne des scripts qui vont
20 modifier ce document à des moments précis. Lorsque de tels scripts ajoutent des éléments, ces éléments sont éliminés lors de la transformation inverse. Les modifications effectuées sur des éléments pré-existants sont par contre plus difficiles à gérer. En effet, il est possible que l'utilisateur écrase ces modifications par des interactions. Par ailleurs lors d'une relecture du document SVG
25 sauvegardé, ces scripts pourront de nouveau modifier le document. Le procédé selon l'invention propose à l'utilisateur pour pallier ce problème d'éditer/modifier les scripts.

Dans de nombreux documents SVG, un script d'initialisation est utilisé pour ajouter de nouveaux éléments graphiques et/ou modifier des
30 éléments présents. Il est important de laisser faire cette initialisation. Pour cette raison, les scripts ne sont pas désactivés au début de l'édition. Les modifications effectuées sont détectées grâce aux événements "mutation" (la création

d'éléments par script peut être détectée autrement) et prises en compte de différentes manières.

5 Par exemple, s'il s'agit de nouveaux éléments, on les marque comme tel (éléments créés par script) et on les place, si ce n'est pas déjà le cas, dans l'élément graphique qui contient l'ensemble des éléments graphiques édités.

Dans un autre exemple, s'il s'agit de modification/d'ajout d'attributs, on indique (via l'espace de nommage des données de gestion) que ces attributs sont modifiés par un script d'initialisation.

10 On autorise certaines modifications des éléments graphiques créés/modifiés par script d'initialisation. Si l'utilisateur souhaite modifier de tels éléments, on le prévient que ces modifications ne seront pas forcément conservées après la phase de sauvegarde. Lorsqu'il s'agit de modifications simples (changement de valeurs d'attributs, d'éléments), on note les
15 modifications sous la forme d'un script et on ajoute à l'élément modifié un attribut (dans l'espace de nommage des données de gestion) qui pointe vers ce nouveau script. La transformation inverse ajoutera ensuite ce script au script appelé à la fin du script d'initialisation.

20 Lors de la transformation inverse, les éléments XML créés par le script d'initialisation sont à enlever. On les étiquette donc comme à enlever.

Le langage SVG supporte certaines fonctionnalités du langage dénommé SMIL, acronyme anglo-saxon pour 'Synchronized Multimedia
Integration Language' et qui est un format d'animation de document multimédia. Ces animations se présentent sous la forme d'éléments `<animate/>`,
25 `<animateMotion/>`, `<animateColor/>` et `<set/>` à l'intérieur d'éléments graphiques. Pour faciliter l'édition de document, la transformation directe possède un paramètre qui permet de définir si l'on souhaite conserver ou désactiver ces éléments, par exemple en les englobant dans un élément dont l'espace de
30 nommage n'est pas celui de SVG mais celui défini pour les informations de gestion. Ainsi, la transformation inverse peut réintégrer ces éléments.

L'utilisateur rentre une URI qui correspond à un document permettant d'exécuter une opération sur tout ou partie du document SVG. Cette URI pointe vers un document du type script (XSLT, JavaScript) ou WSDL

(acronyme anglo-saxon pour « Web Services Description Language ») qui définissent des traitements que l'on peut effectuer sur un fragment de données SVG. L'utilisateur choisit ensuite la partie du document SVG édité à modifier. Le programme lance alors un script qui va permettre d'exécuter le script/le service web. Le script récupère le résultat et l'ajoute au document SVG, éventuellement à la place de la partie du document SVG en entrée.

La transformation directe (étape E110, **figure 1**) initialise les informations utiles pour la transformation inverse (étape E140, **figure 1**). Les scripts ajoutés au document SVG édité permettent de tenir à jour ces informations suivant les interactions utilisateurs. La transformation inverse prend le document SVG modifié par les manipulations de l'utilisateur et le transforme pour la remettre dans un format proche du document SVG édité original. Pour ce faire, la transformation inverse va utiliser ces informations supplémentaires pour construire le document résultat.

En référence à la **figure 5**, à l'étape E500, la transformation inverse récupère le nœud XML à traiter. A l'étape E505, on teste si le nœud a été modifié ou créé par un script d'initialisation. Si oui, on récupère le script de modification s'il en existe un (étape E510), sinon on passe directement à l'étape E515. Cette étape consiste à tester si le nœud est à éliminer. Si c'est le cas on passe à l'étape E520. On récupère alors tous les scripts de modification post-initialisation correspondant aux nœuds enfants que l'on ajoute à la fin du script d'initialisation. On passe ensuite directement au nœud suivant. On peut affiner le processus en séparant les nœuds dont seuls la balise est à enlever, les enfants devant être traités normalement. Dans ce cas, on récupère l'ensemble des enfants et on les met dans la liste des nœuds à traiter.

Si le nœud n'est pas à éliminer, on passe à l'étape E525.

A l'étape E525, on teste une éventuelle modification du nœud par l'utilisateur. En l'absence de modification, on récupère le nœud XML original (étapes E530 et E535) et on l'ajoute au document. Le traitement est alors terminé pour ce nœud et on passe au nœud suivant.

Si le nœud a été modifié ou si l'on ne possède pas le nœud XML original, on passe à l'étape E540. On ajoute un élément correspondant au nœud, on filtre les attributs pour enlever les attributs de l'espace de nommage des

informations de gestion et les attributs d'identification s'ils ont été créés par la transformation directe. Si le nœud a des enfants (étape E545), on traite récursivement chacun des enfants (étape E550), sinon on traite la valeur du nœud (étape E565). Cette étape consiste à copier la valeur sauf dans le cas des

5 nœuds scripts auquel cas on enlève le test correspondant à l'activation/désactivation des scripts. L'algorithme est alors fini (étape E560).

Une fois le document transformé, la dernière étape consiste à intégrer ce document dans un élément <SVG/> auquel on ajoute les informations stockées dans la transformation directe.

10 Par exemple, le document ISVG1 à traiter, en version lecture correspond au document SVG suivant.

```

15      <svg width="250" height="250"
        xmlns="http://www.w3.org/2000/svg"
        xmlns:xlink="http://www.w3.org/1999/xlink"
        xmlns:nsvg="http://example.com/nonsvgns">
          <g      id="Group"      style="stroke-width:4;stroke-
antialiasing:false" >
20              <image xlink:href="image.jpg" x="10" y="10" width="40"
height="40" />
                <text      style="font-size:12;fill-opacity:1"      x="50"
y="10">
25                    My SVG image
                </text>
            </g>
            <script type="text/ecmascript" xlink:href="myscript.js"/>
        </svg>

```

30 En mode écriture, le document ISVG2 correspond aux instructions suivantes.

```

35      <svg width="1600" height="1600" viewBox="0 0 1600 1600"
        xmlns="http://www.w3.org/2000/svg"
        xmlns:xlink="http://www.w3.org/1999/xlink"
        xmlns:nsvg="http://example.com/nonsvgns">
          <g id=" nsvg:svg_image">

```

```

        <!-- l'element suivant encapsule l'image source -->
        <g                                                    id="nsvg:wrapper"
url="http://example.com/servlet/Update?id=10"
        switch="true"            onmouseup="on_mouseup(evt) "
5   onload="init(evt)">
        <g            id="Group"            style="stroke-width:4;stroke-
antialiasing:false" >
                <image      xlink:href="image.jpg"      x="10"      y="10"
width="40" height="40" />
10        <text      style="font-size:12;fill-opacity:1"      x="50"
y="10">
                My SVG image
                </text>
                </g>
15        </g>
        <g nsvg:remove="true">
                <!-- contient les objects UI pour l'interaction et la
visualisation
                contient notamment des GUI pour la selection des
20        objects SVG
                contient notamment des GUI pour la modification
des objects SVG -->
                <text      x="90"      y="10"      style="text-anchor:left;font-
size:12">X=</text>
25        <!-- other svg elements -->
                </g>
                </g>
                <script type="text/ecmascript" nsvg:remove="true">
                <!--nsvg:remove est une info qui va guider la
30        transformation inverse-->
                <![CDATA[
                        function on_mouseup(evt)
                        {
                                // permet l'interaction avec l'utilisateur tel que
35        le drag&drop,
                                // le redimensionnement d'objet...
                        }
                <!-- other javascript functions -->
                ]]>
40        </script>

```



```

        <script type="text/ecmascript" nsvg:href="myscript.js"
nsvg:remove="initThere">
            <!-- inlined scripts functions with a disable
mechanism-->
5         </script>
        </svg>

```

10 Pour cet exemple, les règles de transformation sont définies sous la forme d'un script XSLT. Ce script va notamment inclure des éléments permettant l'interaction avec l'utilisateur (fonction déplacement d'objet graphique utilisant le drag & drop

```

        <xsl:stylesheet version="1.0"
15        xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
        xmlns:xlink="http://www.w3.org/1999/xlink"
        xmlns="http://www.w3.org/2000/svg">
            <xsl:template match="*">
                <svg>
                    <xsl:copy-of select="@*" />
20                <xsl:attribute name="width">1600</xsl:attribute>
                <xsl:attribute name="height">1600</xsl:attribute>
                <xsl:attribute name="viewBox">0 0 1600 1600</xsl:attribute>
                <xsl:if test="not(@id)">
                    <xsl:attribute name="id">root</xsl:attribute>
25                </xsl:if>
                <g id="nsvg:wrapper" switch="true"
onmouseup="on_mouseup(evt) "onload="init(evt) ">
                    <xsl:attribute name="url">...</xsl:attribute>
                    <xsl:for-each select="*[not (name()='script')]">
30                    <xsl:call-template name="transformNode"/>
                    </xsl:for-each>
                    <g nsvg:remove="true">
                        <text x="90" y="10" style="text-anchor:left;font-
size:12">X=</text>
35                    <!-- other svg elements -->
                    </g>
                </g>
                <script type="text/ecmascript" nsvg:remove="true">

```

<!--nsvg:remove est une info qui va guider la transformation inverse-->

```

5         <![CDATA[
            function on_mouseup(evt)
            {
                // permet l'interaction avec l'utilisateur tel
                que le drag&drop,
                // le redimensionnement d'objet...
            }
10        <!-- other javascript functions -->
        ]]>

        </script>
        <!--other svg elements-->
        </svg>
15    </xsl:template>
    <xsl:template name="transformNode">
        <-- transformation rules for each svg child -->
        <xsl:template>
20    </xsl:stylesheet>

```

Par ailleurs les règles d'écriture sont ici par exemple du type suppression, insertion, modification ou analogue.

En référence à la **figure 6**, on a représenté un dispositif mettant en oeuvre le logiciel permettant de traiter les documents SVG conformément au procédé de l'invention.

Le dispositif peut être par exemple un micro-ordinateur 10 connecté à différents périphériques, par exemple, une caméra numérique 107 (ou un scanner, ou tout moyen d'acquisition ou de stockage d'image) reliée à une carte graphique et fournissant des informations à traiter selon l'invention.

Le dispositif 10 comporte une interface de communication 112 reliée à un réseau 113 apte à transmettre des informations numériques. Le dispositif 10 comporte également un moyen de stockage 108 tel que par exemple un disque dur. Il comporte aussi un lecteur de disquette 109. La disquette 110 comme le disque 108 peuvent contenir des données d'implantation logicielle de l'invention ainsi que le code de l'invention qui, une fois lu par le dispositif 10, sera stocké dans le disque dur 108. Selon une variante, le programme permettant au

dispositif de mettre en oeuvre l'invention, pourra être stocké en mémoire morte 102 (appelée ROM sur le dessin). Il en est de même pour les méthodes de codage. En seconde variante, le programme pourra être reçu pour être stocké de façon identique à celle décrite précédemment par l'intermédiaire du réseau de communication 113.

Le dispositif 10 est relié à un microphone 111 par l'intermédiaire de la carte E/S 106. Les données à traiter selon l'invention seront dans ce cas du signal audio.

Ce même dispositif possède un écran 104 permettant de visualiser les informations à traiter ou de servir d'interface avec l'utilisateur qui pourra paramétrer certains modes de traitement, à l'aide du clavier 114 ou de tout autre moyen (souris par exemple).

L'unité centrale 100 (appelée CPU) exécute les instructions relatives à la mise en oeuvre de l'invention, instructions stockées dans la mémoire morte 102 ou dans les autres éléments de stockage. Lors de la mise sous tension, les programmes et méthodes de traitement stockés dans une des mémoires non volatile, par exemple la ROM 102, sont transférés dans la mémoire vive RAM 103 qui contiendra alors le code exécutable de l'invention ainsi que les variables nécessaires à la mise en oeuvre de l'invention. En variante, les méthodes de traitement pourront être stockées dans différents endroits. En effet, il est possible d'améliorer l'invention en ajoutant de nouvelles méthodes transmises soit par le réseau de communication 113 ou par l'intermédiaire de disquette 110. Bien entendu, les disquettes peuvent être remplacées par tout support d'information tel que CD-ROM ou carte mémoire.

Le bus de communication 101 permet la communication entre les différents sous éléments du micro-ordinateur 10 ou liés à lui. La représentation du bus 101 n'est pas limitative et notamment l'unité centrale 100 est susceptible de communiquer des instructions à tout sous élément de 10 directement ou par l'intermédiaire d'un autre sous élément du micro-ordinateur 10.

Le dispositif décrit ici est susceptible de contenir tout ou partie du traitement décrit dans l'invention.

REVENDEICATIONS

5 1. Procédé de traitement d'au moins un document graphique numérique représenté en un langage de balisage prédéterminé dans lequel au moins un logiciel d'affichage de type butineur est utilisé pour afficher un tel document, caractérisé en ce que le procédé comprend les étapes suivantes :

10 i) transformer le document d'origine (ISVG1) affiché en mode lecture en une version éditable (ISVG2) dans le langage de balisage selon un jeu de règles de transformation prédéfinies non liées au document qui incorporent un jeu de règle d'écriture au document,

ii) interagir via le logiciel d'affichage avec ladite version éditable (ISVG2) pour la modifier selon ledit jeu de règles d'écriture, et

15 iii) transformer la version éditable ainsi modifiée (ISVG2) en une version en mode lecture (ISVG3) incorporant les modifications apportées lors de l'étape ii).

20 2. Procédé selon la revendication 1, caractérisé en ce que le procédé comprend en outre l'étape iv) dans laquelle il est prévu d'afficher ou stocker ladite version modifiée en mode lecture (ISVG3).

25 3. Procédé selon la revendication 1 ou la revendication 2, caractérisé en ce que le logiciel d'affichage de type butineur est capable d'utiliser le jeu de règles de transformation pour transformer le document d'origine en une version éditable.

30 4. Procédé selon l'une des revendications 1 à 2, caractérisé en ce que le jeu de règles de transformation est mis sous le format de scripts XSLT et/ou Javascript.

5. Procédé selon l'une des revendications 1 à 2, caractérisé en ce que le jeu de règles de transformation est proposé sous le format d'un service web.



6. Procédé selon l'une des revendications 1 à 5, caractérisé en ce que le jeu de règles d'écriture appartient au groupe formé par des fonctions d'insertion, de modification, d'élimination, de déplacement, et de copie d'au moins certains éléments ou nœuds du document à éditer.

5

7. Procédé selon l'une quelconque des revendications précédentes, caractérisé en ce que la transformation inverse selon l'étape iii) est apte, à partir d'un document édité (ISVG2), à récupérer le document non édité (ISVG3).

10

8. Procédé selon l'une quelconque des revendications précédentes, caractérisé en ce que la transformation directe selon l'étape i) est apte à ajouter des informations de guidage susceptibles de guider la transformation inverse selon l'étape iii).

15

9. Procédé selon la revendication 8, caractérisé en ce que les informations de guidage appartiennent au groupe formé par des éléments à éliminer ; des éléments se trouvant dans le document modifié (ISVG2) dans un espace de nommage spécifique ; des scripts mettant à jour les valeurs des informations de guidage ; des informations d'indication relatives à la création/modification des attributs.

20

10. Procédé selon l'une quelconque des revendications précédentes, caractérisé en ce que la transformation directe selon l'étape i) est apte à identifier chaque élément graphique sélectionnable.

25

11. Procédé selon l'une quelconque des précédentes revendications, caractérisé en ce que la transformation directe selon l'étape i) est apte à englober/faire passer des nœuds écrits dans le langage de balisage de type SVG dans un espace de nommage non SVG pour désactiver le effets desdits nœuds.

30

12. Procédé selon la revendication 11, caractérisé en ce que la transformation inverse selon l'étape iii) est apte à récupérer les nœuds SVG passés dans un espace de nommage non SVG.

5 13. Procédé selon l'une quelconque des revendications précédentes, caractérisé en ce que la transformation directe selon l'étape i) comprend un paramètre susceptible de décider de conserver/d'éliminer un élément d'animation.

10 14. Procédé selon l'une quelconque des revendications précédentes, caractérisé en ce que la transformation directe selon l'étape i) incorpore des événements de mutation aptes à modifier de façon synchrone le document (ISVG2) par rapport au document initial (ISVG1).

15 15. Procédé selon l'une quelconque des revendications précédentes, caractérisé en ce que la transformation directe selon l'étape i) est apte à visualiser dynamiquement le document édité (ISVG2).

20 16. Procédé selon l'une quelconque des revendications précédentes, caractérisé en ce que la transformation directe selon l'étape i) incorpore un mécanisme apte à modifier tout ou partie du document édité (ISVG2) via des programmes disponibles à distance du document.

25 17. Procédé selon la revendication 16, caractérisé en ce que lesdits programmes appartiennent au groupe formé par des scripts JavaScript ; des scripts XSLT ; des services Web.

30 18. Procédé selon la revendication 15 ou la revendication 16, caractérisé en ce que lesdits programmes sont aptes à prendre en entrée un arbre écrit en langage de balisage de type SVG et retourner une image non vectorielle en sortie.



19. Procédé selon l'une quelconque des revendications précédentes, caractérisé en ce que la transformation inverse selon l'étape iii) est apte à modifier un script d'initialisation pour sauvegarder des modifications effectuées sur des éléments graphiques créés par ledit script d'initialisation.

5

20. Dispositif de traitement d'au moins un document graphique numérique représenté en un langage de balisage prédéterminé dans lequel au moins un logiciel d'affichage de type butineur est utilisé pour afficher un tel document caractérisé en ce qu'il comprend :

10

- des moyens de transformation directe pour transformer le document d'origine (ISVG1) affiché en mode lecture en une version éditable (ISVG2) dans le langage de balisage selon un jeu de règles de transformation prédéfinies non liées au document qui incorporent un jeu de règle d'écriture au document ;

15

- des moyens de traitement pour interagir via le logiciel d'affichage avec ladite version éditable (ISVG2) pour la modifier selon ledit jeu de règles d'écriture ; et

20

- des moyens de transformation inverse pour transformer la version éditable ainsi modifiée (ISVG2) en une version en mode lecture (ISVG3) incorporant les modifications ainsi apportées par les moyens de traitement.

21. Dispositif selon la revendication 20, caractérisé en ce qu'il comprend en outre des moyens pour afficher ou stocker ladite version modifiée en mode lecture (ISVG3).

25

22. Dispositif selon la revendication 20 ou la revendication 21, caractérisé en ce que le jeu de règles d'écriture appartient au groupe formé par des fonctions d'insertion, de modification, d'élimination, de déplacement, et de copie d'au moins certains éléments ou nœuds du document à éditer.

30

23. Dispositif selon l'une quelconque des revendications 20 à 22, caractérisé en ce que les moyens de transformation inverse sont aptes à récupérer à partir d'un document édité (ISVG2), le document non édité (ISVG3).

24. Dispositif selon l'une quelconque des revendications précédentes 20 à 23, caractérisé en ce que les moyens de transformation directe sont aptes à ajouter des informations de guidage susceptibles de guider les moyens de transformation inverse.

5

25. Dispositif selon la revendication 24, caractérisé en ce que les informations de guidage appartiennent au groupe formé par des éléments à éliminer ; des éléments se trouvant dans le document modifié (ISVG2) dans un espace de nommage spécifique ; des scripts mettant à jour les valeurs des informations de guidage ; des informations d'indication relatives à la création/modification des attributs.

10

26. Dispositif selon l'une quelconque des revendications 20 à 25, caractérisé en ce que les moyens de transformation directe sont aptes à identifier chaque élément graphique sélectionnable.

15

27. Dispositif selon l'une quelconque des revendications 20 à 26, caractérisé en ce que les moyens de transformation directe sont aptes à englober/faire passer des nœuds écrits dans le langage de balisage de type SVG dans un espace de nommage non SVG pour désactiver le effets desdits nœuds.

20

28. Dispositif selon la revendication 27, caractérisé en ce que les moyens de transformation inverse sont aptes à récupérer les nœuds SVG passés dans un espace de nommage non SVG.

25

29. Dispositif selon l'une quelconque des revendications 20 à 28, caractérisé en ce que les moyens de transformation directe sont adaptés pour utiliser un paramètre susceptible de décider de conserver/d'éliminer un élément d'animation.

30

30. Dispositif selon l'une quelconque des revendications 20 à 29, caractérisé en ce que les moyens de transformation directe sont adaptés pour



traiter des évènements en mutation aptes à modifier de façon synchrone le document (ISVG2) par rapport au document initial (ISVG1).

5 31. Dispositif selon l'une quelconque des revendications 20 à 30, caractérisé en ce que les moyens de transformation directe sont aptes à visualiser dynamiquement le document édité (ISVG2).

10 32. Dispositif selon l'une quelconque des revendications 20 à 31, caractérisé en ce que les moyens de transformation directe comprennent un mécanisme apte à modifier tout ou partie du document édité (ISVG2) via des programmes disponibles à distance du document.

15 33. Dispositif selon la revendication 32, caractérisé en ce que lesdits programmes appartiennent à au groupe formé par des scripts JavaScript ; des scripts XSLT ; des services Web.

20 34. Dispositif selon la revendication 32 ou la revendication 33, caractérisé en ce que lesdits programmes sont aptes à prendre en entrée un arbre écrit en langage de balisage de type SVG et retrouver une image non vectorielle en sortie.

25 35. Dispositif selon l'une quelconque des revendications 20 à 34, caractérisé en ce que les moyens de transformation inverse sont aptes à modifier un script d'initialisation pour sauvegarder des modifications effectuées sur des éléments graphiques créés par ledit script d'initialisation.

30 36. Support d'informations lisible par un système informatique, éventuellement amovible, totalement ou partiellement, notamment CD-ROM ou support magnétique, tel un disque dur ou une disquette, ou support transmissible, tel un signal électrique ou optique, caractérisé en ce qu'il comporte des instructions d'un programme d'ordinateur permettant la mise en œuvre du procédé selon l'une quelconque des revendications 1 à 19, lorsque ce programme est chargé et exécuté par un système informatique.

37. Programme d'ordinateur stocké sur un support d'informations, ledit programme comportant des instructions permettant la mise en œuvre d'un procédé de traitement selon l'une quelconque des revendications 1 à 19, lorsque
5 le programme est chargé et exécuté par un système informatique.

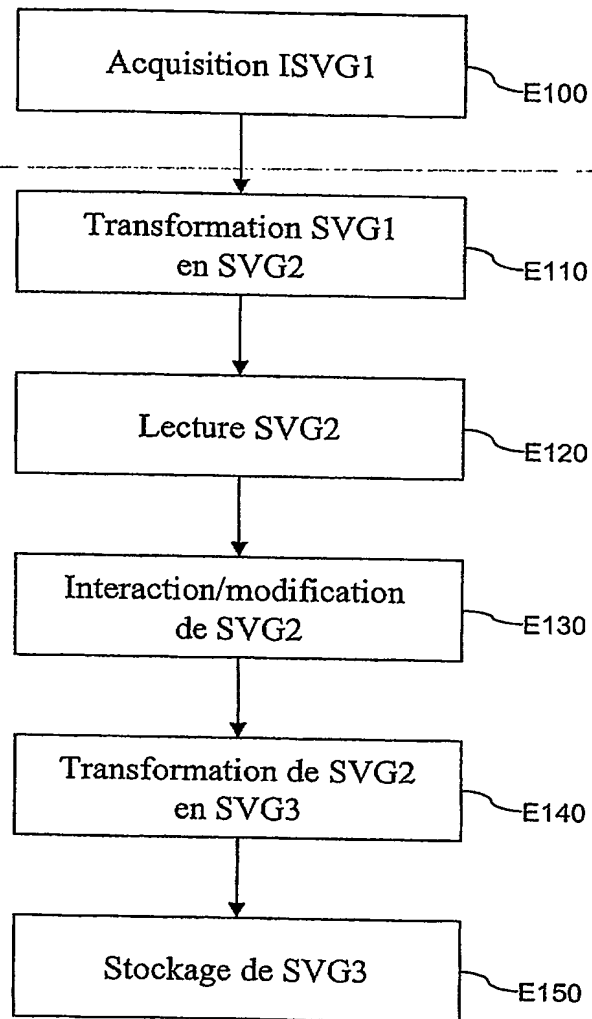
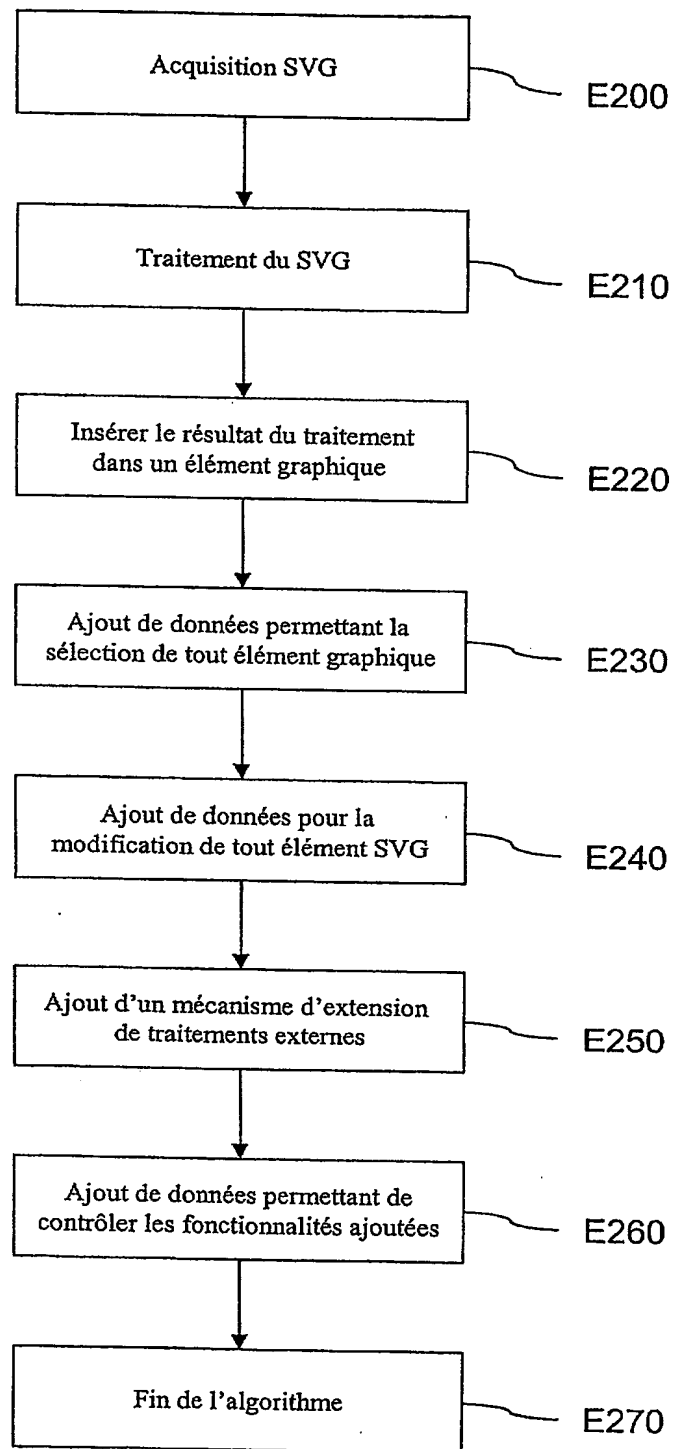


Figure 1

**Figure 2**

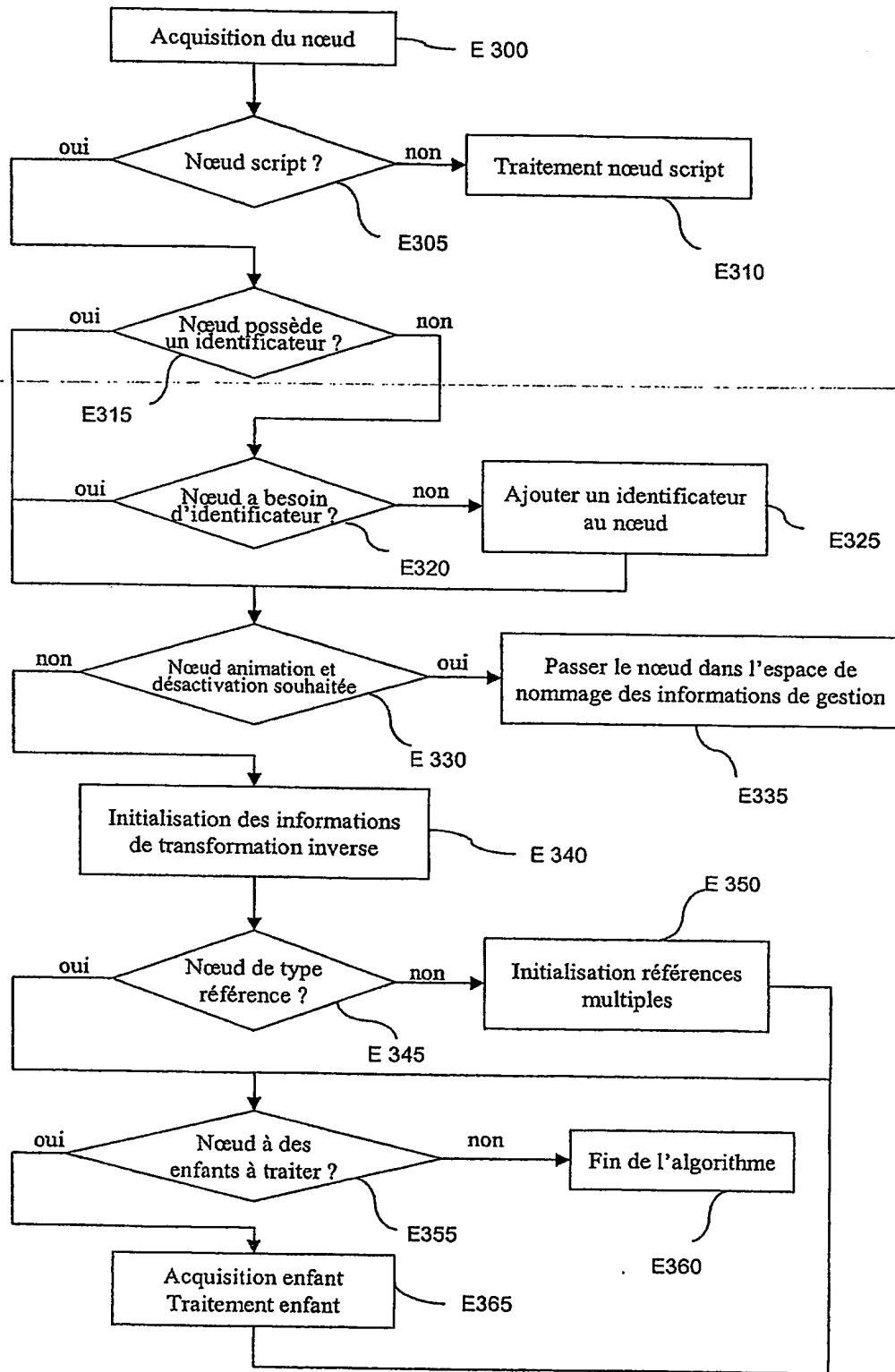
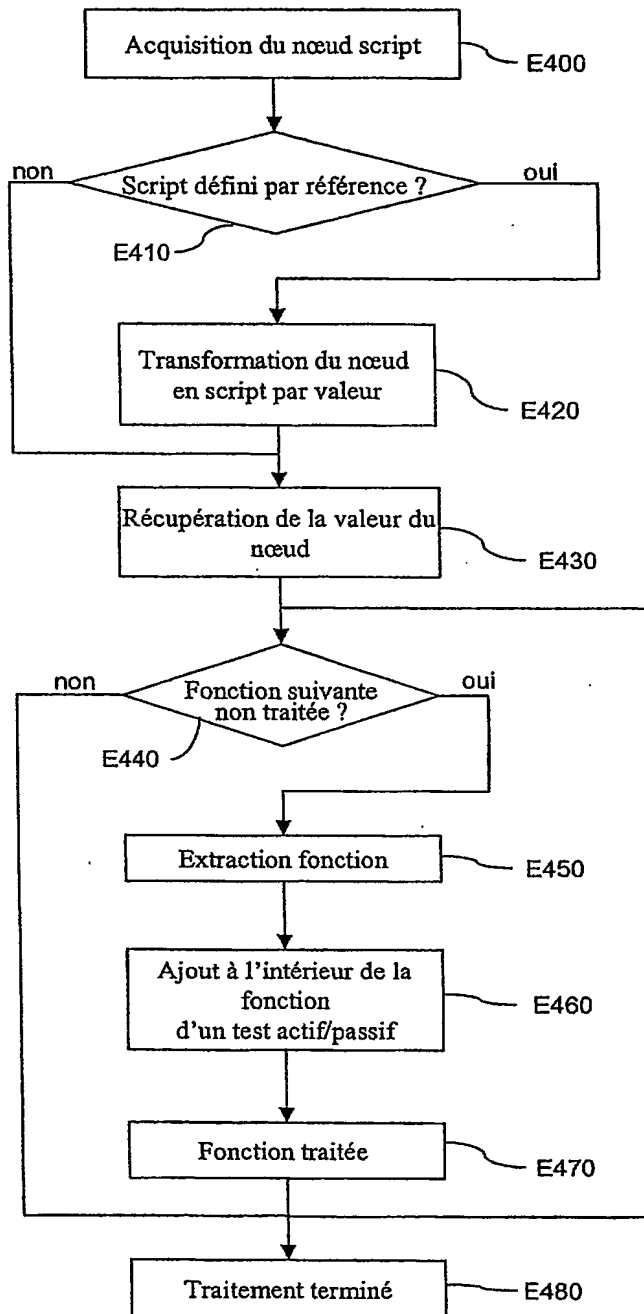


Figure 3

**Figure 4**

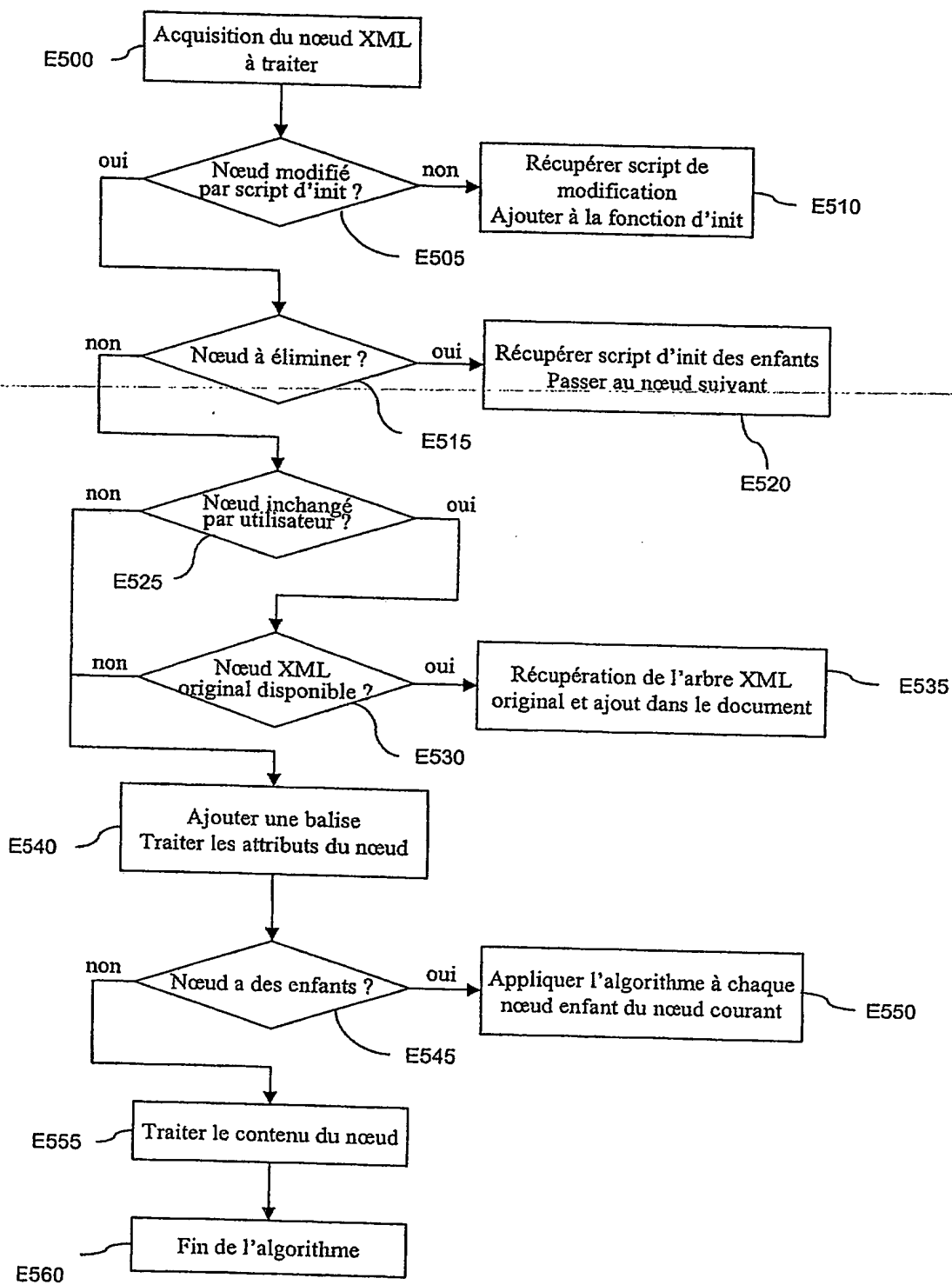
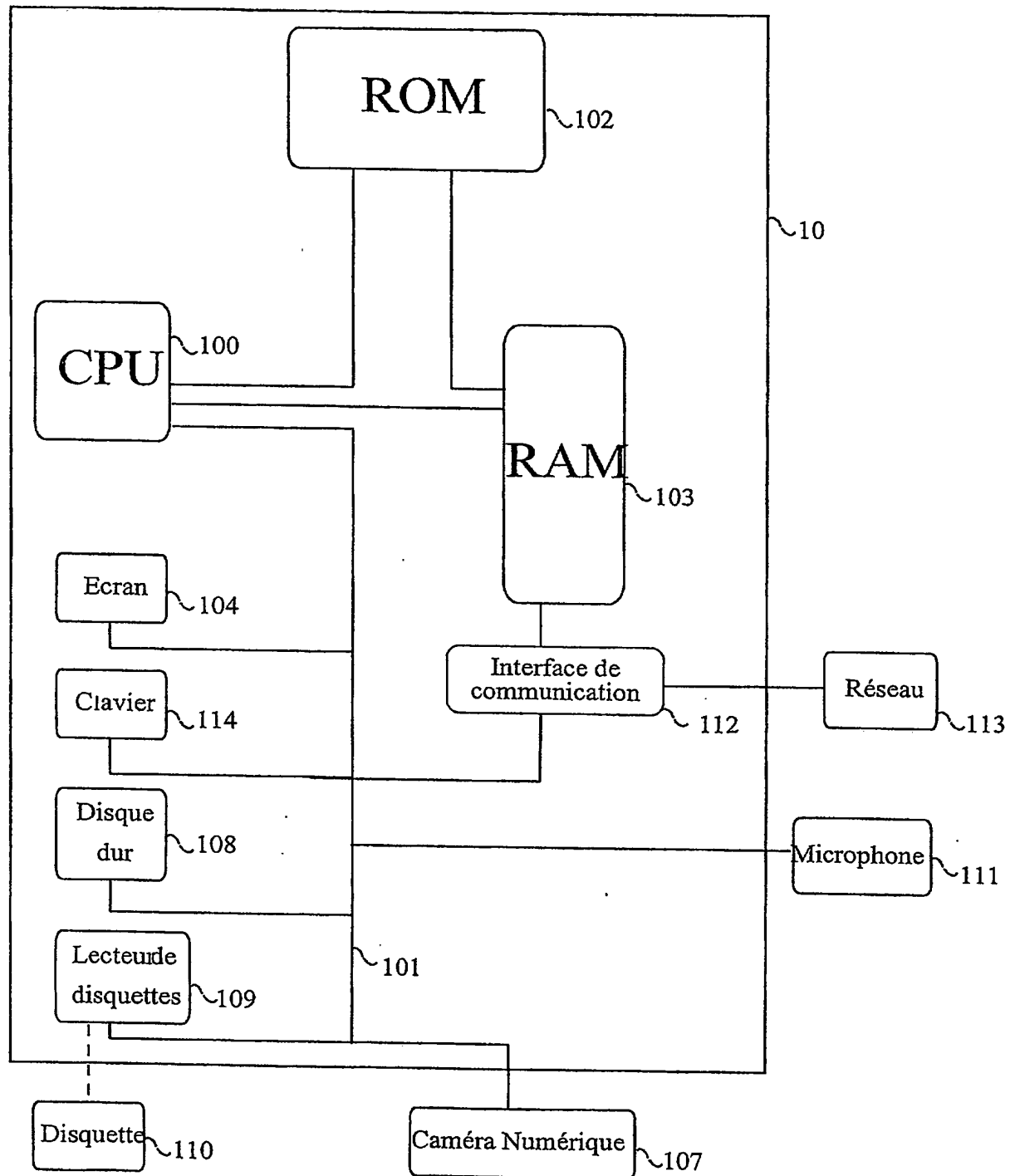


Figure 5

**Figure 6**



BREVET D'INVENTION

CERTIFICAT D'UTILITÉ

Code de la propriété intellectuelle - Livre VI

N° 11235*03

DÉPARTEMENT DES BREVETS

26 bis, rue de Saint Pétersbourg
75800 Paris Cedex 08

Téléphone : 33 (1) 53 04 53 04 Télécopie : 33 (1) 42 94 86 54

DÉSIGNATION D'INVENTEUR(S) Page N° 1../1..

(À fournir dans le cas où les demandeurs et les inventeurs ne sont pas les mêmes personnes)

INV

Cet imprimé est à remplir lisiblement à l'encre noire

DB 113 @ W / 270501

Vos références pour ce dossier (facultatif)		BIF116008/ON/LJH
N° D'ENREGISTREMENT NATIONAL		03 11 436
TITRE DE L'INVENTION (200 caractères ou espaces maximum) Procédé et dispositif d'édition de documents graphiques numériques du type SVG notamment à partir d'un butineur.		
LE(S) DEMANDEUR(S) : CANON KABUSHIKI KAISHA		
DESIGNE(NT) EN TANT QU'INVENTEUR(S) :		
1 Nom		FABLET
Prénoms		Youenn
Adresse	Rue	Le Pâtis de la Grette
	Code postal et ville	3 5 3 9 0 La DOMINELAIS, France
Société d'appartenance (facultatif)		
2 Nom		
Prénoms		
Adresse	Rue	
	Code postal et ville	
Société d'appartenance (facultatif)		
3 Nom		
Prénoms		
Adresse	Rue	
	Code postal et ville	
Société d'appartenance (facultatif)		
S'il y a plus de trois inventeurs, utilisez plusieurs formulaires. Indiquez en haut à droite le N° de la page suivi du nombre de pages.		
DATE ET SIGNATURE(S) DU (DES) DEMANDEUR(S) OU DU MANDATAIRE (Nom et qualité du signataire) Le 30 septembre 2003 Olivier NICOLLE N°92.3040 SANTARELLI		

La loi n°78-17 du 6 janvier 1978 relative à l'informatique, aux fichiers et aux libertés s'applique aux réponses faites à ce formulaire. Elle garantit un droit d'accès et de rectification pour les données vous concernant auprès de l'INPI.

**This Page is Inserted by IFW Indexing and Scanning
Operations and is not part of the Official Record**

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☒ **BLACK BORDERS**
- ☐ **IMAGE CUT OFF AT TOP, BOTTOM OR SIDES**
- ☐ **FADED TEXT OR DRAWING**
- ☐ **BLURRED OR ILLEGIBLE TEXT OR DRAWING**
- ☐ **SKEWED/SLANTED IMAGES**
- ☐ **COLOR OR BLACK AND WHITE PHOTOGRAPHS**
- ☐ **GRAY SCALE DOCUMENTS**
- ☐ **LINES OR MARKS ON ORIGINAL DOCUMENT**
- ☐ **REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY**
- ☐ **OTHER:** _____

IMAGES ARE BEST AVAILABLE COPY.

As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.